MULTİTHREAD İLE ASANSÖR SİMÜLASYONU

Alparslan Beraat Özdemir - 170202045
Bilgisayar Mühendisliği Bölümü
Kocaeli Üniversitesi
Kocaeli, Türkçe
beraat78@gmail.com

ÖZET

Bu, AVM'de çalışan 5 asansörün eşzamanlı çalışmasını simüle eden bir programdır. AVM içerisindeki katlarda bulunun insanlardan asansör kullanmak isteyenlerin yoğunluğu artınca, otomatik olarak yoğunluk durumuna göre bir veya daha fazla diğer asansörler aktif hale gelir.

I. GİRİŞ

AVM Asansör Simülasyonu, tek bir class içerisinde kodlanmıştır. Program Multi-Thread bir yapıya sahiptir. 5 adet asansör Threadi, 1 adet kontrol, 1 adet giriş ve 1 adet de çıkış Threadi olmak üzere toplamda 8 threadden oluşmaktadır.

Asansör threadleri hariç diğer threadler timer ile oluşturulmuştur. Arayüz olarak konsolda yapılan her işlem log kaydı gibi akacak şekilde ayarlanmıştır.

Her Thread, yaptığı işlemi biter bitmez ekrana yazdırmaktadır.

Program bir kere çalıştırıldığında, kullanıcı tarafından el ile müdahale edilerek durdurulmadığı sürece aralıksız olarak çalışmaya devam etmektedir.

Program çalışır çalışmaz bütün threadler başlatılır.

Birinci asansör hariç diğer bütün asansörlerin çalışma durumu kontrol Threadi tarafından yönetilir.

II. TEMEL BİLGİLER

Proje gelişiminde;

Tümleşik geliştirme ortamı olarak Apache Netbeans IDE 12.0 kullanılmıştır.

Program Windows 10 İşletim sisteminde test edilmiştir. Program Java dilinde geliştirilmiştir.

III. TASARIM

Asansör Simülasyonunun programlanma aşamaları altta belirtilen başlıklar altında açıklanmıştır.

A. Veri yapılarını oluşturmak.

Program multi-thread olduğu için, asansör kuyruklarını tutacak veri yapısı olarak Thread-safe olması açısından ConcurrentLinkedQueue isimli kuyruk veri yapısı kullanılmıştır. Her asansör için 10 boyutlu Integer arrayler tanımlanarak asansörün yolcuları burada tutması sağlanmıştır.

B. Threadler

Avmgiris Threadi:

Bu Thread, düzenli aralıklarla avmnin giriş katındaki kuyruğun sonuna 1-10 arasında, rastgele katlara gitmek isteyen insanlar eklemektedir. AVM'deki insan sirkülasyonu için olmazsa olmaz olan bu thread, her 500 ms'de bir bir timer ile çağırılarak çalışmaktadır.

Avmcikis Threadi:

Bu Thread de Timer üzerinde çalışmaktadır. Her 1000 MS aralıkta bir kez çalışarak, zemin kat hariç 2-3-4-5. Katlardan 1-5 arasında rastgele insanın avmden çıkış yapmak için kat kuyruklarına eklenmesini sağlamaktadır.

Bu Thread, katlardaki insan sayılarını kontrol eder ve eğer yeterli insan yoksa, katlarda bir miktar insan birikmesini bekleyerek kuyruğa kimseyi almaz.

Kuyrukkontrol Threadi:

Bu Thread, katlardaki kuyrukların yoğunluğunu kontrol eder. Eğer katlardaki yoğunluk, aktif asansör sayısının

toplam kapasitesinin 2 katından 1 fazla ise, bir asansörü daha aktif eder. Eğer yoğunluk bu oranın altına düşerse, 1 asansör kalana kadar asansörleri sırayla devre dışı bırakır.

Asansor Threadleri:

Asansör Threadleri, temel olarak asansörlerin çalışmasını sağlayan threadlerdir. Toplam 5 tanedir. Asansor1 threadi hariç diğer 4 tanesi kontrol threadine bağlıdır ve onun tarafından kontrol edilir.

Pasif durumda iken Threadler durdurulmaz, çalışmaya devam ederler fakat genel olarak uyku halindedirler. Hiçbir işlem yapmazlar.

Asansörler eğer dağıtım modunda iseler, ki genelde aşağıdan yukarı çıkarken dağıtım modunda olurlar, hiçbir katta yolcu almaz, sadece içlerindeki yolcuları gitmek istedikleri katlara dağıtırlar.

İçlerinde yolcu kalmadığı zaman toplama moduna geçerler. Toplama modunda ise kuyruğu dolu olan en üst kata çıkarak o kattan itibaren AVM'den çıkmak isteyen insanları asansöre alırlar.

Zemin kata her geldiklerinde içlerindeki tüm yolcuları boşaltır ve tekrar, giriş kattan AVM'ye girmek isteyen yolcuları alır ve dağıtım moduna geçerler.

C. Algoritma

1- Başla
2- Avmgiris Threadini başlat. Sonsuza kadar belli aralıklarla çalışır.
3- Avmcikis Threadini başlat. Sonsuza kadar belli aralıklarla çalışır.
4- Kuyrukkontrol Threadini başlat. Sonsuza kadar belli aralıklarla çalışır.
5- Asansör Threadlerini başlat. Sonsuza kadar çalışırlar.
6- Avmgiris Threadi ile AVM'ye 1-10 arasında yolcu al.
7- Bu yolcuları giriş kat asansör kuyruğuna ekle.
8- Yolcuları Asansör Threadi ile asansöre al.
9- Asansör Threadini dağıtım moduna geçir.
10- Yolcuları gitmek istedikleri katlarda indir.
11- Katlara inen müşteri sayısını ekle.
12- Asansör boşaldıysa toplama moduna geç.
13- Bekleyen yolcu olan en üst kattan itibaren yolcuları toplamaya başla.
14- Zemin katta indir. Tekrarla.
15- Avmcikis Threadi Katları kontrol eder.
16- Eğer katlarda müşteri varsa, bu müşterilerden rastgele 1-5 tanesini avmcikis için kat kuyruğuna al.
17- Müşteri yoksa hiçbir şey yapma.

IV. SONUÇLAR VE EKRAN ÇIKTILARI

Program ilk çalıştığında asansör1'in yaptığı ilk turun sonuçları şekil 1'de gösterilmiştir.

```
run:
Asansor1: asansor kat 1 de.
Asansor1: Zemin katta cikan kisi sayisi: 0
Asansor1: Yeni yolcular zemin kattan Asansore aliniyor...
Asansor1: alinanlar= 2
Asansor1: alinanlar= 5
Asansor1: asansor kat 2 de.
Asansor1: asansor dagitimda.
Asansor1: kat2 de yolcu indirildi.
Asansor1: kat2 de yolcu indirme islemi basariyla tamamlandi. sonraki kata gidiliyor.
Asansor1: asansor kat 5 de.
Asansor1: asansor dagitimda.
Asansor1: kat5 de yolcu indirildi.
Asansor1: kat5 de yolcu indirme islemi basariyla tamamlandi. sonraki kata gidiliyor.
Asansor1: yolcu kalmadi. 5. kattan itibaren toplama modu aktif.
```

Şekil1.

Avmcikis threadinin katlardan yolcuları kat kuyruklarına alması şekil 2'de belirtilmiştir.

```
Asansor1: yolcu kalmadi. 5. kattan i
----Kat3'den 1 kisi kuyruga alindi.
----Kat4'ten 1 kisi kuyruga alindi.
----Kat5'ten 1 kisi kuyruga alindi.
```

Şekil 2.

Programın multi-Thread çalışması ve kontrol Threadinin çalıştığı şekil 3'te gösterilmiştir.

```
asansor3: kat 3 de.
Asansor3: 3. kattaki bir yolcu alindi.
asansor4: kat 3 de.
Asansor2: kat 3 de.
Asansor1: asansor kat 3 de.
asansor4: asansor4 kat 2 de.
Asansor2: asansor2 kat 2 de.
asansor3: asansor3 kat 2 de.
Asansor1: asansor kat 2 de.
ASANSOR 4 DEVREDISI.....
----Kat5'ten 1 kisi kuyruga alindi.
asansor4: kat 1 de.
Asansor2: kat 1 de.
```

şekil 3.

KAYNAKÇA

[1] https://docs.oracle.com/javase/7/docs/api/java/util/concurrent/ConcurrentLinkedQueue.html

[2] https://serdarkuzucu.com/javada-timer-ve-timertask-siniflarinin-kullanimi/

[3] https://ufukuzun.wordpress.com/2015/02/14/javada-multithreading-bolum-1-merhaba-thread/